NAME
rrdbuild – Instructions for building RRDtool

OVERVIEW
If you downloaded the source of RRDtool you have to compile it. This document will give some information on how this is done.

RRDtool relies on services of third part libraries. Some of these libraries may already be installed on your system. You have to compile copies of the other ones before you can build RRDtool.

This document will tell you about all the necessary steps to get going.

These instructions assume you are using a bash shell. If you use csh/tcsh, then you can either type bash to switch to bash for the compilation or if you know what you are doing just replace the export bits with setenv.

We further assume that your copies of tar and make are actually GNU tar and GNU make respectively. It could be that they are installed as gtar and gmake on your system.

OPTIMISTIC BUILD
Before you start to build RRDtool, you have to decide two things:

1. In which directory you want to build the software.
2. Where you want to install the software.

Once you have decided. Save the two locations into environment variables.

BUILD_DIR=/tmp/rrdbuild
INSTALL_DIR=/opt/rrdtool−1.7.2

If your /tmp is mounted with the option noexec (RHEL seems to do that) you have to choose a different directory!

Now makes sure the BUILD_DIR exists and go there:

mkdir −p $BUILD_DIR
cd $BUILD_DIR

Lets first assume you already have all the necessary libraries pre-installed.

wget https://oss.oetiker.ch/rrdtool/pub/rrdtool−1.7.2.tar.gz
gunzip −c rrdtool−1.7.2.tar.gz | tar xf −
cd rrdtool−1.7.2
./configure --prefix=$INSTALL_DIR && make && make install

Ok, this was very optimistic. This try will probably have ended with configure complaining about several missing libraries.

INSTALLING DEPENDENCIES
If your OS lets you install additional packages from a software repository, you may get away with installing the missing packages. When the packages are installed, run configure again and try to compile again. Below you find some hints on getting your OS ready for compiling RRDtool.

Additions to this list are welcome. In general RRDtool should work with the latest versions of the libraries. The versions listed here are just what was current when I tested this.

OpenSolaris 2008.05
Just add a compiler and the gnome development package:

pkg install sunstudioexpress
pkg install SUNWgnome−common−devel

There is a problem with cairo.pc on OpenSolaris. It suggests that xrender is required for compilation with cairo. This is not true and also bad since OpenSolaris does not include an xrender.pc file. Use Perl to fix this:
perl -i -p -e 's/(Requires.*?)\s*xrender.*$/1/' /usr/lib/pkgconfig/cairo.pc

Make sure the RRDtool build system finds your new compiler

```bash
export PATH=/opt/SunStudioExpress/bin
```

**Debian / Ubuntu**

Use apt-get to make sure you have all that is required. A number of packages will get added through dependencies.

```bash
apt-get install libpango1.0-dev libxml2-dev
```

**Gentoo**

In Gentoo installing RRDtool is really simple you just need to `emerge rrdtool`. All dependencies will be handled automatically by the portage system. The only thing you should care about are USE flags, which allow you fine tune features RRDtool will be built with. Currently the following USE flags are available:

- `doc` - install .html and .txt documentation into /usr/share/doc/rrdtool-1.x.xx/
- `perl` - build and install perl language bindings
- `python` - build and install python language bindings
- `ruby` - build and install ruby language bindings
- `tcl` - build and install tcl language bindings
- `rrd cgi` - build and install rrdcgi

After you’ve decided which USE flags you need, set them either in `make.conf` or `/etc/portage/package.use` and finally run:

```bash
# emerge -va rrdtool
```


**BUILDING DEPENDENCIES**

But again this may have been too optimistic still, and you actually have to compile your own copies of some of the required libraries. Things like libpng and zlib are pretty standard so you will probably have them on your system anyway. Freetype, Fontinst, Cairo, Pango may be installed, but it is possible that they are pretty old and thus don’t live up to our expectations, so you may want to compile their latest versions.

**General build tips for AIX**

If you are working with AIX, you may find the `--disable-shared` option will cause things to break for you. In that case you may have to install the shared libraries into the RRDtool PREFIX and work with `--disable-static` instead.

Another hint to get RRDtool working on AIX is to use the IBM XL C Compiler:

```bash
export CC=/usr/vac/bin/cc
export PERLCC=$CC
```

(Better instructions for AIX welcome!)

**Build Instructions**

Some libraries want to know where other libraries are. For this to work, set the following environment variable

```bash
export PKG_CONFIG_PATH=${INSTALL_DIR}/lib/pkgconfig
export PATH=${INSTALL_DIR}/bin:$PATH
```

The above relies on the presence of the `pkgconfig` program. Below you find instructions on how to compile pkgconfig.

Since we are compiling libraries dynamically, they must know where to find each other. This is done by setting an appropriate LDFLAGS. Unfortunately, the syntax again differs from system to system:
Solaris

```bash
export LDFLAGS=-R${INSTALL_DIR}/lib

if you are using the Sun Studio/Forte compiler, you may also want to set
CFLAGS="-xO3 -xcode=pic13"  (SPARC)
CFLAGS="-xO3 -Kpic"  (x86)
```

Linux

```bash
export LDFLAGS="-Wl,--rpath -Wl,${INSTALL_DIR}/lib"
```

HPUX

```bash
export LDFLAGS="+b${INSTALL_DIR}/lib"
```

AIX

```bash
export LDFLAGS="-Wl,-blibpath:${INSTALL_DIR}/lib"
```

If you have GNU make installed and it is not called 'make', then do

```bash
export MAKE=gmake
export GNUMAKE=gmake
```

otherwise just do

```bash
export MAKE=make
```

Building pkgconfig

As mentioned above, without pkgconfig the whole build process will be lots of pain and suffering, so make sure you have a copy on your system. If it is not available natively, here is how to compile it.

```bash
wget http://pkgconfig.freedesktop.org/releases/pkg-config-0.23.tar.gz
gunzip -c pkg-config-0.23.tar.gz | tar xf -
cd pkg-config-0.23
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC"
$MAKE
$MAKE install
```

After installing pkgconfig in a custom directory, setting up the corresponding environment variable will be helpful.

```bash
export PKG_CONFIG=$INSTALL_DIR/bin/pkg-config
```

Building zlib

Chances are very high that you already have that on your system ...

```bash
cd $BUILD_DIR
wget https://oss.oetiker.ch/rrdtool/pub/libs/zlib-1.2.3.tar.gz
gunzip -c zlib-1.2.3.tar.gz | tar xf -
cd zlib-1.2.3
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC" --shared
$MAKE
$MAKE install
```

Building libpng

Libpng itself requires zlib to build, so we need to help a bit. If you already have a copy of zlib on your system (which is very likely) you can drop the settings of LDFLAGS and CPPFLAGS. Note that the backslash (\) at the end of lines means that the command is split over multiple lines.
cd $BUILD_DIR
wget https://oss.oetiker.ch/rrdtool/pub/libs/libpng-1.2.18.tar.gz
gunzip -c libpng-1.2.18.tar.gz | tar xf -
cd libpng-1.2.18
env CFLAGS="-O3 -fPIC" ./configure --prefix=$INSTALL_DIR
$MAKE
$MAKE install

Building freetype

cd $BUILD_DIR
wget https://oss.oetiker.ch/rrdtool/pub/libs/freetype-2.3.5.tar.gz
gunzip -c freetype-2.3.5.tar.gz | tar xf -
cd freetype-2.3.5
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC"
$MAKE
$MAKE install

If you run into problems building freetype on Solaris, you may want to try to add the following at the start the configure line:
env EGREP=egrep

Building LibXML2

cd $BUILD_DIR
wget https://oss.oetiker.ch/rrdtool/pub/libs/libxml2-2.6.32.tar.gz
gunzip -c libxml2-2.6.32.tar.gz | tar xf -
cd libxml2-2.6.32
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC"
$MAKE
$MAKE install

Building fontconfig

Note that fontconfig has a run time configuration file in INSTALL_DIR/etc you may want to adjust that so that fontconfig finds the fonts on your system. Run the fc-cache program to build the fontconfig cache after changing the config file.

cd $BUILD_DIR
wget https://oss.oetiker.ch/rrdtool/pub/libs/fontconfig-2.4.2.tar.gz
gunzip -c fontconfig-2.4.2.tar.gz | tar xf -
cd fontconfig-2.4.2
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC" --with-freetype-config=$INSTALL_DIR/bin/freetype-config
$MAKE
$MAKE install

Building Pixman

cd $BUILD_DIR
wget https://oss.oetiker.ch/rrdtool/pub/libs/pixman-0.10.0.tar.gz
gunzip -c pixman-0.10.0.tar.gz | tar xf -
cd pixman-0.10.0
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC"
$MAKE
$MAKE install

Building Cairo
cd $BUILD_DIR
grep https://oss.oetiker.ch/rrdtool/pub/libs/cairo-1.6.4.tar.gz
gunzip -c cairo-1.6.4.tar.gz | tar xf -
    cd cairo-1.6.4
./configure --prefix=$INSTALL_DIR
    --enable-xlib=no 
    --enable-xlib-render=no 
    --enable-win32=no 
    CFLAGS="-O3 -fPIC"
$MAKE
$MAKE install

When building on Solaris you may want to do

./configure --prefix=$INSTALL_DIR
    --enable-xlib=no 
    --enable-xlib-render=no 
    --enable-win32=no 
    CFLAGS="-O3 -fPIC -D_POSIX_PTHREAD_SEMANTICS"

Building Glib

cd $BUILD_DIR
grep https://oss.oetiker.ch/rrdtool/pub/libs/glib-2.15.4.tar.gz
gunzip -c glib-2.15.4.tar.gz | tar xf -
    cd glib-2.15.4
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC"
$MAKE
$MAKE install

Building Pango

cd $BUILD_DIR
bunzip2 -c pango-1.21.1.tar.bz2 | tar xf -
    cd pango-1.21.1
./configure --prefix=$INSTALL_DIR CFLAGS="-O3 -fPIC" --without-x
$MAKE
$MAKE install

Building rrdtool (second try)

Now all the dependent libraries are built and you can try again. This time you tell configure where it should be looking for libraries and include files. This is done via environment variables. Depending on the shell you are running, the syntax for setting environment variables is different.

And finally try building again. We disable the python and tcl bindings because it seems that a fair number of people have ill configured python and tcl setups that would prevent RRDtool from building if they are included in their current state.

cd $BUILD_DIR/rrdtool-1.7.2
./configure --prefix=$INSTALL_DIR --disable-tcl --disable-python
$MAKE clean
$MAKE
$MAKE install

SOLARIS HINT: if you want to build the Perl module for the native Perl (the one shipping with Solaris) you will need the Sun Forte compiler installed on your box or you have to hand-tune bindings/perl-shared/Makefile while building!

Now go to $INSTALL_DIR/share/rrdtool/examples/ and run them to see if your build has been successful.
AUTHOR
Tobias Oetiker <tobi@oetiker.ch>